**ii) 32 Bit addition**

**DATA SEGMENT**

**NUM1 DW 0FFFFH,0FFFFH**

**NUM2 DW 1111H,1111H**

**SUM DW 4 DUP(0)**

**DATA ENDS**

**CODE SEGMENT**

**ASSUME CS:CODE,DS:DATA**

**START: MOV AX,DATA**

**MOV DS,AX**

**MOV AX,NUM1**

**ADD AX,NUM2**

**MOV SUM,AX**

**MOV AX,NUM1+2**

**ADC AX,NUM2+2**

**;Move LSB of NUM1 to AX**

**;Add LSB of NUM2 to AX**

**;Store the LSB in SUM**

**; Move MSB of NUM1 to AX**

**; Add MSB of NUM2 to AX**

**JNC DOWN ; Check for carry**

**MOV SUM+4,01H**

**DOWN: MOV SUM+2,AX**

**MOV AH,4CH**

**INT 21H**

**CODE ENDS**

**END START**

**INPUT: 0FFFFFFFFH, 011111111H**

**OUTPUT: 0111111110H**

**iv) 16 Bit Subtraction**

**DATA SEGMENT**

**NUM DW 4567H,2345H**

**DIF DW 1 DUP(0)**

**DATA ENDS**

**CODE SEGMENT**

**ASSUME**

**CS:CODE,DS:DATA**

**START: MOV AX,DATA**

**MOV DS,AX**

**CLC**

**; Clearing Carry**

**LEA SI,NUM ; SI pointed to the NUM**

**MOV AX,[SI] ; Move NUM1 to AX**

**SBB AX,[SI+2] ; Move the SI to Num2 and subtract with AX(Takes**

**;care for both smaller as well as larger**

**;Number subtraction)**

**MOV DIF,AX ;Store the result**

**MOV AH,4CH**

**INT 21H**

**CODE ENDS**

**END START**

**v) 32 Bit Subtraction**

**DATA SEGMENT**

**NUM1 DW 2345H,6762H**

**NUM2 DW 1111H,1111H**

**DIF DW 2 DUP(0)**

**DATA ENDS**

**CODE SEGMENT**

**ASSUME CS:CODE,DS:DATA**

**START: MOV AX,DATA**

**MOV DS,AX**

**LEA SI,NUM1**

**LEA DI,NUM2**

**MOV AX,[SI]**

**MOV BX,[DI]**

**SUB AX,BX**

**MOV DIF,AX**

**INC SI**

**; SI pointed to the LSB of NUM1**

**; DI pointed to the LSB of NUM2**

**; Move the content of SI to AX**

**; Move the content of DI to BX**

**; Subtract from BX to AX**

**; Store the LSB result in DIF**

**;Update SI to point the MSB of NUM1(if**

**;ADD SI,02 instruction its affect carry flag)**

**INC SI**

**INC DI**

**INC DI**

**MOV AX,[SI]**

**MOV BX,[DI]**

**;Update DI to point the MSB of NUM2**

**; Move the content of SI to AX**

**; Move the content of DI to BX**

**SBB AX,BX ; Subtract with borrow from BX to AX**

**MOV DIF+2,AX**

**MOV AH,4CH**

**INT 21H**

**CODE ENDS**

**END START**

**INPUT: 23456762,-11111111**

**OUTPUT:12345651**

**INPUT:11111111,-23451234**

**OUTPUT:EDCBFEDD**

**vi)16 Bit multiplication for unsigned numbers**

**DATA SEGMENT**

**NUM DW 1234H,1234H**

**PROD DW 2 DUP(0)**

**DATA ENDS**

**CODE SEGMENT**

**ASSUME CS:CODE,DS:DATA**

**START: MOV AX,DATA**

**MOV DS,AX**

**LEA SI,NUM ; SI pointed to the Multiplicand**

**MOV AX,[SI] ; Multiplicand has to be in AX register**

**MOV BX,[SI+2] ; SI+2 pointed to the Multiplier and move it to BX**

**MUL BX ;Perform the multiplication**

**MOV PROD,AX ;32 bit product stored in DX-AX registers**

**MOV PROD+2,DX**

**MOV AH,4CH**

**INT 21H**

**CODE ENDS**

**END START**

**INPUT: Multiplicand- 1234H,**

**Multiplier - 1234H**

**OUTPUT: DX-01 4B**

**AX-54 90**

**vii)16 Bit multiplication for signed numbers**

**DATA SEGMENT**

**NUM DW -2,1**

**PROD DW 2 DUP(0)**

**DATA ENDS**

**CODE SEGMENT**

**ASSUME CS:CODE,DS:DATA**

**START: MOV AX,DATA**

**MOV DS,AX**

**LEA SI,NUM ; SI pointed to the Multiplicand**

**MOV AX,[SI] ; Multiplicand has to be in AX register**

**MOV BX,[SI+2] ; SI+2 pointed to the Multiplier and move it to BX**

**IMUL BX ; Perform the sign multiplication using sign**

**;Multiplication operator (IMUL)**

**MOV PROD,AX ; 32 bit product stored in DX-AX registers**

**MOV PROD+2,DX**

**MOV AH,4CH**

**INT 21H**

**CODE ENDS**

**END START**

**INPUT: Multiplicand- -2,**

**Multiplier - 1**

**OUTPUT: DX – FF FF**

**AX – FF FE ; Result is in two complement form**